IKW'99 – 1st International Khepera Workshop, Paderborn, Alemanha, dezembro de 1999.

A Genetic Based Navigator System

Marco Aurélio C. Pacheco[†‡], Marley Maria B. R. Vellasco[†‡], Laercio B. Gonçalves[†], and Carlos E. Thomaz[†]

[†]ICA: Laboratório de Inteligência Computacional Aplicada, Departamento de Engenharia Elétrica
Pontifícia Universidade Católica do Rio de Janeiro – PUC/Rio, Brazil

[‡]Departamento de Engenharia de Sistemas e Computação
Universidade do Estado do Rio de Janeiro – UERJ, Brazil

E-mail: ica@ele.puc-rio.br

Abstract. Genetic Algorithms (GAs) mimic models of natural evolution and have the ability to adaptively search large spaces in near-optimal ways. In this paper we describe a GA learning system that evolves chromosomes representing the control attitudes of the Khepera robot. These attitudes define the basic robot actions to reach a goal location, performing straight motion and avoiding obstacles. The GA fitness function, employed to teach robot's movements, was engineered to achieve this type of behavior in spite of any changes in Khepera's goals and environment. The results obtained demonstrate the controller's adaptability, displaying near-optimal paths in different configurations of the environment.

1 Introduction

Genetic Algorithms (GAs) are powerful procedures for searching large, complex and multi-modal spaces [6,9]; a number of researchers have employed them to solve autonomous agent path planning problems [3,13]. Brooks [5] proposed to use an extension of Koza's genetic programming technique [11], evolving explicit programs. Hoffmann [10] described a messy genetic algorithm for the automatic design of fuzzy logic controllers. Dorigo and Schnepf [7] proposed to use a form of classifier system. Others proposed to evolve neural networks [1,4,8]. More recently, Jing Xiao and Lixin Zhang [14] combined the concept of evolutionary computation with problem-specific chromosome path structures and operators.

In this article, we present a approach to navigation problem consisting of a GA controller system that evolves chromosomes representing the robot's attitudes through a learning process. These attitudes define the basic motor speed actions and have a fitness function engineered to teach a simulated mobile robot Khepera to reach the goal location, performing straight motion and avoiding obstacles. The learning process does not depend on the obstacle environment distribution, nor on the initial and final navigation locations. Therefore, this GA-based controller doesn't need to be retrained when changes on these parameters occur.

2 The Genetic Model

The genetic model presented in this paper is a new approach to the path-planning problem, representing each chromosome as a group of basic robot's attitudes. These attitudes define the robot's movements in agreement with the feedback generated by its environment. Each feedback, which is used as input to the system, is based on the sensors reading and on the

robot's direction to its goal location (Figure 1).

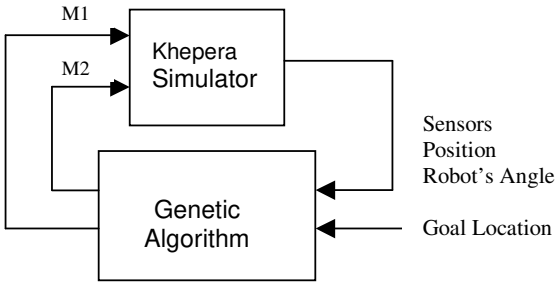


Figure 1. The Genetic Navigation System

The sensors reading is presented to the GA system in a simplified form. Although the simulator provides 8 sensors values, this number could be reduced without affecting Khepera's performance [2].

The other input to the system, the robot's direction to its goal location, is calculated as follows. Consider α as the direction's angle of the robot, provided by the simulator at any Khepera's location. Then, define β as the angle of minimum distance between robot's location and its fixed arrival position. Thus, the difference between β and α indicates if the goal point is in front of the robot, behind it, to its left or right sides, determining one of the possible four Khepera's target direction. Figure 2 and Figure 3 illustrates these considerations.

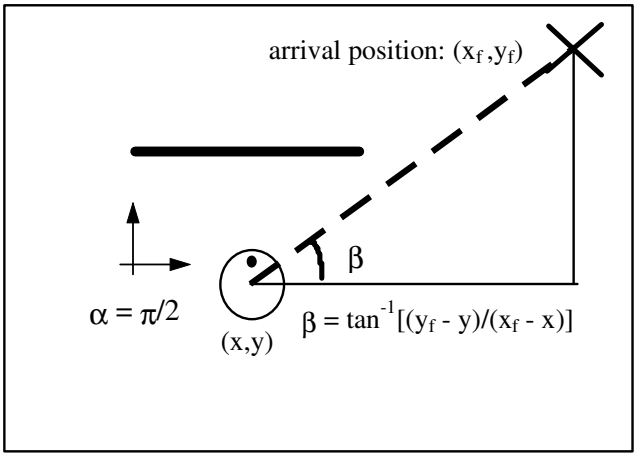


Figure 2. The angle β points to the minimum goal's distance direction.

Figure 2 shows the robot looking to the front at $\alpha=\pi/2$ degrees. The angle β is calculated by the trigonometric function $\tan^{-1}$ between Khepera (x,y) and arrival (x_f, y_f) positions. In this example, β can be supposed equal to $\pi/4$ degrees. Thus, the difference $\beta - \alpha$ is equal to $-\pi/4$ and this value indicates that the goal point is located at Khepera's right side (see Figure 3(A)). If, for instance, Khepera has been looking to the right at $\alpha=0$ degrees, the difference $\beta - \alpha$ would be equal to $\pi/4$ and its target direction would point to the robot's left side.

In order to check our model, we have designed a second model in which is an improved version of the first, willing to achieve higher performance in the tests. In the second direction model (Figure 3(B)) we have considered 8 possible target directions. With that new model,

we expect that the Khepera will be more precise in reaching the final target.

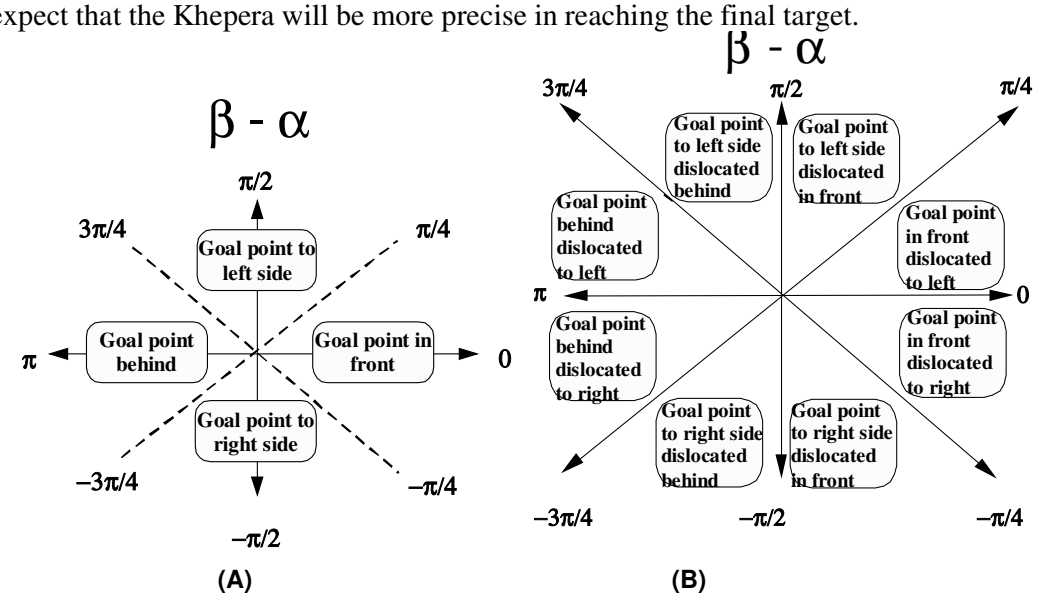


Figure 3. Target direction of Khepera's movement. (A) First Direction Model: four possible Khepera's target direction. (B) Second Direction Model: four possible Khepera's target direction.

From the sensors reading and the robot's direction inputs, for both models, the Khepera's attitude can be defined according to the following states rule:

IF ((S_{left} > L) or (S_{right} > L) or (S_{back} > L))

THEN

Obstacle detected, avoid collision, forget target

Proximity-sensor = highest value (S_{left}, S_{right}, S_{back})

ELSE

Obstacle not detected (collision-free), straight to the target

Target direction = β - α

END

The constant L represents a collision threshold and may be determined between the sensor's range [0, 1023].

Each possible attitude defined by the states rule corresponds to a gene of the chromosome, and each gene is composed of a pair of Khepera's motor speeds (M1,M2). The representation chromosome representation for the first direction model has the form presented in Figure 4.

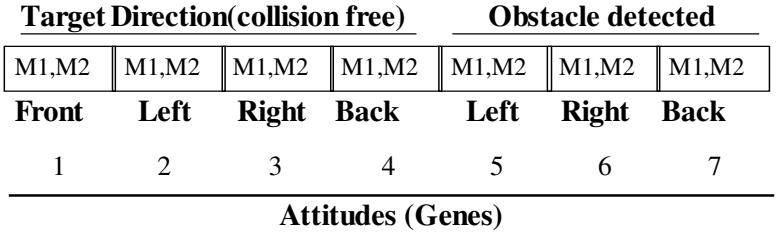


Figure 4. 7-Genes chromosome representation.

In the first direction model the representation, composed by 7 genes, divides the chromosome in two parts. The first one determines the basic robot's target direction movements when no obstacles are detected (collision free states). The second part (obstacle detected states) considers the motor speed actions that avoid collision, being independent from the goal's location. Thus, if the robot's state is defined as collision free and its goal point is in its left side ($\pi/4 \leq \beta-\alpha \leq 3\pi/4$), then the Khepera's attitude to be carried out corresponds to the second gene of the chromosome. On the other hand, if Khepera detects the proximity of an object at its right hand side, it is going to take the action indicated at gene 6, without considering the target direction.

For the second direction model, the chromosome representation is a generalization of the first model, as shown in Figure 5.

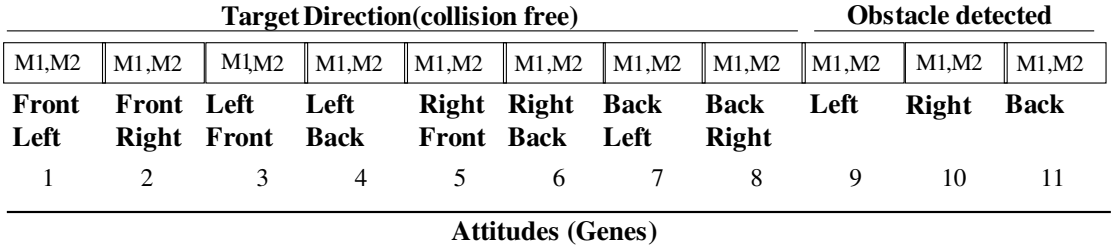


Figure 5. Eleven Genes chromosome representation.

The fitness function F is calculated at each robot's time step and is composed of three components: V, D and A (see Figure 6).

$$F = \sum_{i=1}^{\#genes} (Vi * Di * Ai); 0 \leq Vi \leq 1; 0 \leq Di \leq 1; 0 \leq Ai \leq 1$$

	7-gene chromosome	11-gene chromosome
$Vi = (\lvert M1 \rvert + \lvert M2 \rvert)/(2 * M\max)$		
$Di = \begin{cases} 1 \\ (1 + ((M1 + M2)/(2 * M\max)))/2 \end{cases}$	,i = [2,3,4] ,i = [1,5,6,7]	,i = [3,4,5,6,7,8] ,i = [1,2,9,10,11]
$Ai = \begin{cases} 0 & ,TPi = 0 \\ (\sum_{t=1}^{TPi} AAi_t) \Big/ TPi & ,TPi \neq 0 \end{cases}$		
$AAi_t = \begin{cases} \Delta di/d\max \\ \Delta gi/g\max \\ 1 - S_t/S\max \end{cases}$	,if (i = 1) and (Δdi ≤ 0) ,else $AAi_t = 0$,if (i = [2,3,4]) and (Δgi ≤ 0) ,else $AAi_t = 0$,if i = [5,6,7]	,if (i = [1,2]) and (Δdi ≤ 0) ,else $AAi_t = 0$,if (i = [3,4,5,6,7,8]) and (Δgi ≤ 0) ,else $AAi_t = 0$,if i = [9,10,11]

Figure 6. The fitness function for 7-genes and 11-genes chromosomes.

Where:

- $|M|$: absolute value of speed ranging [-10,10];
- i : attitude's index;
- Vi : attitude's speed ;
- Di : attitude's straight motion ;
- Ai : attitude's action;
- TPi : total of steps executed by attitude i ;
- t : step index;
- AAi_t: action's fitness at step t;
- Δdi : difference of goal's distance between two consecutive steps;
- Δgi : difference of target's direction between two consecutive steps;
- S_t : proximity-sensor (S_{left}, S_{right}, S_{back}) highest activity at step t;
- dmax : maximum possible distance traveled by robot in only one step (limited by the simulator);
- gmax : maximum possible robot's angle rotation in only one step (limited by the simulator);
- Mmax : maximum robot's speed (equal to 10);
- Smax : maximum sensor's reading (equal to 1023).

For the case of the 7-gene chromosome, the first component (V) is maximized by speed, the second (D) by straight motion and the third (A) by action. The latter depends on the attitude defined according to the states rule. If Khepera's state is collision free and the arrival point is at its front (attitude 1), then the genetic algorithm (GA) should evolve the pair M1,M2 that minimizes this distance. On the other hand, if the robot is free but not oriented to the goal point (attitudes 2, 3 and 4), then it should find the pair (M1,M2) that rotates the Khepera's front in the direction of the target. It should be observed that in these attitudes (2,3 and 4) the straight direction component (D) is made equal to 1 to avoid contradictory learning. Finally, if an object is located in the range of Khepera's view (attitudes 5,6 and 7), then GA should forget the goal location during some steps and find the best way (pair M1,M2) to avoid possible collision. The action component (A) is calculated by the average of the total steps executed per attitude because of the random noise (around 10%) added by the simulator in the amplitudes of the motor speed and sensors reading [12].

For the case of 11-gene chromosome, the fitness function F is the same for the case of 7-gene chromosome, with differences only in the components D e AAi_t, as seen in Figure 6.

The GA program was performed as follows. An initial population of individuals was created by assigning to each gene in the chromosomes a random new speed value ranging [-10,10]. The Khepera was left free to move as a result of the activity generated by the gene's attitudes, while its performance was recorded and accumulated according to the pre-designed fitness function. Each individual could move for a limited number of steps and all their starting conditions were made the same.

Three genetic operators - *selective reproduction, one-point crossover, and mutation* - were applied to create new populations of the same size. Selective reproduction consists of a linear

scaling of the fitness values followed by a probabilistic allocation of a number to the offspring proportional to the fitness value of each chromosome. All offsprings, simple copies of their parents, were then randomly paired and a random one-point crossover was performed with a given probability. Each value of the newly obtained strings was then mutated with a given probability by adding a small random value within a negative and positive mutation speed range ([-10,10]). Thus, in this evolutionary system, the new population replaces the old one maintaining only the best individual (elitism). The process terminates after a number of generations, and the best chromosome represents the near-optimum attitudes found.

3 Evaluation Experiments

In order to evaluate the system's performance, many tests were carried out, varying the GA's parameters. The best performance was achieved with the following parameters : population size equal to 100 individuals; crossover rate set to 80%; mutation rate set to 4%; constant L of collision threshold equal to 900; and total number of steps (n) equal to 700. The total number of steps (n parameter) was defined as bigger as possible, in order to test all robot's states without compromising the processing time.

The system takes approximately one minute to evaluate each generation of 100 individuals. Furthermore, around the 50th generation the best individual usually dominates the population and further evolution can not be noticed. Thus, 25 runs of 50 generations were executed, saving the best chromosome of each run. Then, the best chromosome over all these runs was used to guide the robot in different environments, starting at different initial locations and with distinct goal points. The total number of steps for the robot reaching the target point were computed. Figure 7 shows the best chromosomes obtained for the case of 7-gene and 11-gene chromosome models.

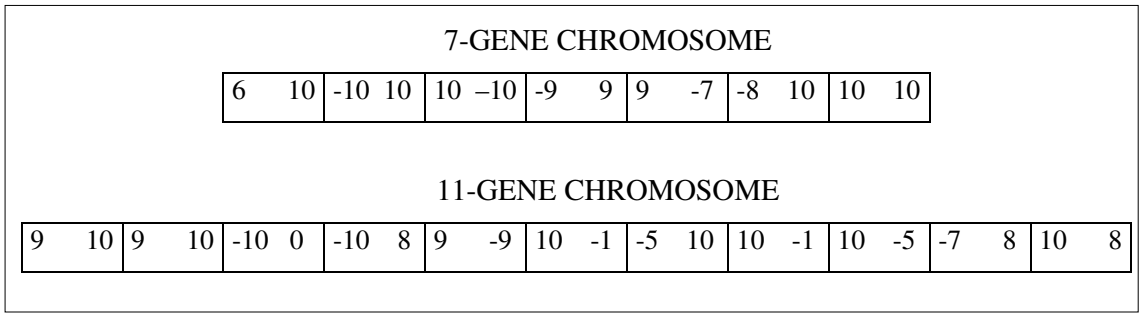
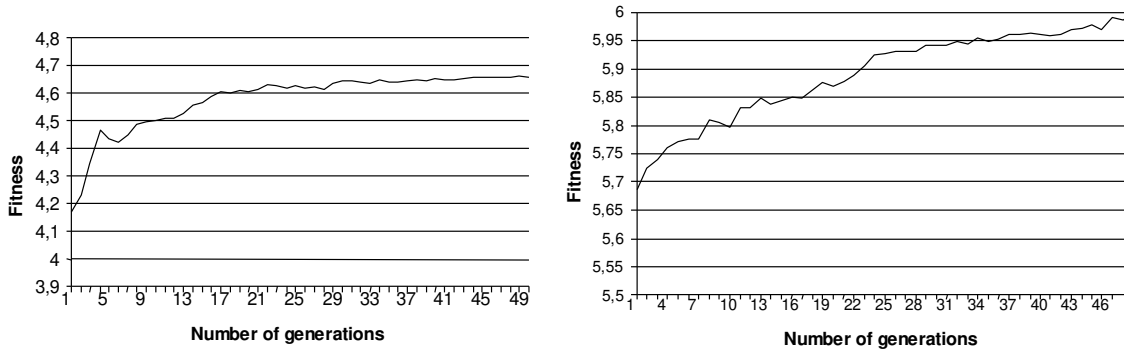


Figure 7. Best chromosomes obtained for the case of 7-gene chromosome and for the case of 11-gene chromosome.

For both models, we have obtained the graphics of the GA's average performance, corresponding to the average of best chromosomes in 25 experiments (Figure 8). As it can be seen in these graphics, the 11-gene model clearly outperforms the 7-gene model in terms of fitness, thus confirming our expectation.



(A) (B)

Figure 8. (A) Average of the best chromosome in 25 experiments for the case of 7 genes chromosome. (B) Average of the best chromosome in 25 experiments for the case of 11 genes chromosome.

The best chromosomes were tested in several environments. The trajectory made by the robot, in several situations can be observed in Figure 9 and Figure 10, where target and initial position of the robot are set to different points in the environment.

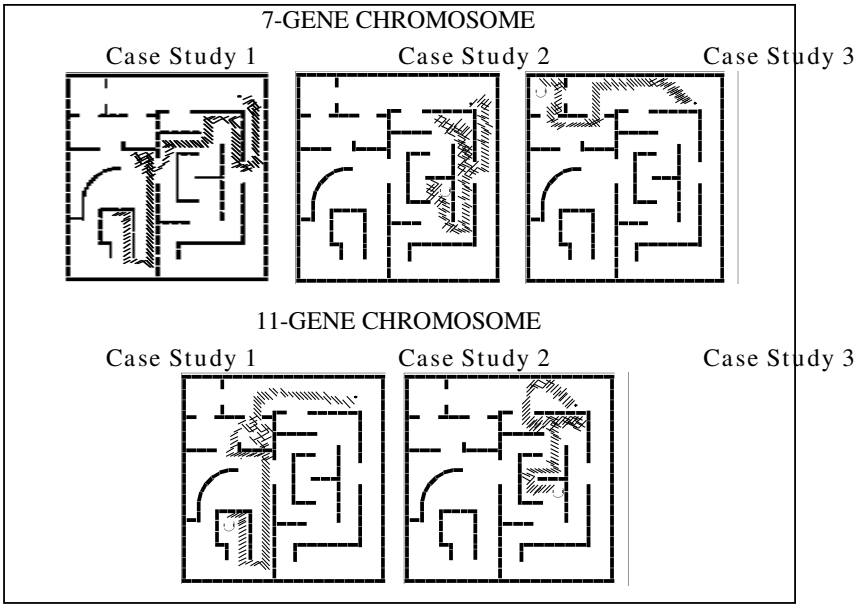


Figure 9. Paths Achieved by Khepera with the 7-gene chromosome and with the 11-gene chromosome, for the three cases studies.

Table 1 and Table 2 present a speed comparison between the 7-gene chromosome and the 11-gene chromosome models for the case studies show in Figure 9 and Figure 10, respectively. As it can be seen, the Khepera is clearly much faster with the 11-gene chromosome model than with the 7-gene chromosome model. However, there are situations in which the robot could not reached the target point, regardless the chromosome model. The reason is that the robot has no memory about previous unsuccessful paths and may get stuck in trick environments.

Speed Comparison		
	7 Genes Chromosome	11 Genes Chromosome
Case Study 1	4160	2209
Case Study 2	3611	1141
Case Study 3	1184	905

Table 1. Speed comparison between the case of 7-gene chromosome and the case of 11-gene chromosome, for the three cases studies show by Figure 9.

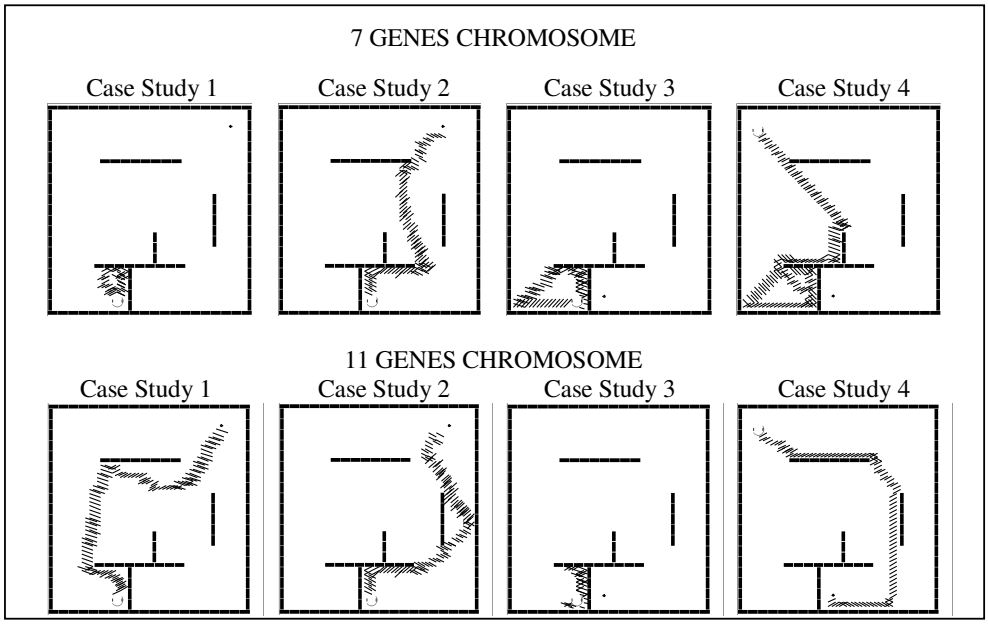


Figure 10. Paths Achieved by Khepera with the 7-gene chromosome and with the 11-gene chromosome, for four cases of studies.

Speed Comparison		
	7 Genes Chromosome	11 Genes Chromosome
Case Study 1	Target not reached	729
Case Study 2	1124	618
Case Study 3	Target not reached	Target not reached
Case Study 4	Target not reached	1655

Table 2. Speed comparison between the case of 7-gene chromosome and the case of 11-gene chromosome, for the three cases studies show by Figure 10.

Overall, the Khepera tends to move towards the goal's location, traveling with the higher speed in theirs motors. When the robot detects an obstacle, it forgets the target's direction during some steps and tries to avoid possible collision. Soon after this is solved and Khepera becomes collision free, it again looks for the goal's point, moving towards the minimum distance to the arrival location. In this experiment, as soon as Khepera perceives an object, it avoids the collision moving away from the objects, although these actions increase its distance from the arrival location.

4 Conclusions

The experiments presented in this paper confirm the power of Genetic Algorithms (GAs) for solving multi-criterion optimization problems. Most importantly, unlike various learning techniques, GA can gradually evolve the control system of an autonomous robot by exploiting the variations in the interactions between the environment and the robot itself. This makes it possible to apply them to problems where neither the shape nor the size of the solution is known in advance.

The results obtained showed that the use of GA for the Khepera's controller produced extremely smooth path planning. The robot usually travels at full motor speed, successfully finds the diverse goal points from distinct starting points and environments and, at the same time, avoids obstacle and walls. It should also be observed that the robot's learning process does not depend on the environment, so that the robot doesn't need to be retrained if changes in starting point, goal point, and obstacle distribution occur.

The fitness function chosen had an important role in the success of this system. The fitness function was engineered in order to maximize multiple targets: the fast and straight navigation; the collision avoidance; and the location of the goal point. This was possible because these sub-goals do not represent contradicted meanings, allowing equal normalization. However, the strategy of evaluating each attitude (gene) of the chromosome in an independent way became limited in some situations. This limitation was characterized in some cases where the robot reached its target location after a clearly exaggerated total number of steps, getting almost stuck. This can be justified by the fact that Khepera has been modeled to forget its goal location if it encounters a collision dangerous area (area with obstacles) and, consequently, being able to move away from its arrival point in such cases. Also, the fitness functions corresponding to collision detection attitudes have not been well refined, analyzing only the highest activity sensor whatever side it belongs to.

The results show that the robot "learned" how to achieve the target avoiding collisions. The controller improved performance with the 11-gene, thus validating the genetic model employed (chromosome representation, fitness function and operators). Other tasks can be included in the model (e.g. energy supply) although, chromosome codification is limited for few robot's situations.

Acknowledgment. This work was supported by CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), and CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico), Brazil.

References

[1] **K. Balakrishnan, and V. Honavar,** Analysis of neurocontrollers designed by simulated evolution. *Proceedings of the International Conference on Neural Networks*, Washington, D.C., 1996.

[2] **K. Balakrishnan, and V. Honavar,** On sensor evolution in robotics. *Proceedings of the First International Conference on Genetic Programming*, Standford University, CA, pp. 455-460, 1996.

[3] **K. Balakrishnan, and V. Honavar,** Some Experiments in Evolutionary Synthesis of Robotic Neurocontrollers. *Proceedings of the World Congress on Neural Networks (WCNN'96)*, San Diego, CA, September 15-20, pp. 1035-1040, 1996.

[4] **S. Baluja,** Evolution on artificial neural networks based autonomous land vehicle controller, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 26, no. 3, pp. 450-463, June 1996.

[5] **R. A. Brooks,** Artificial life and real robots. Towards a Practice of Autonomous Systems: *Proceedings of the First European Conference on Artificial Life* edited by F. J. Varela, P. Bourgine. Cambridge, MA : MIT Press/Bradford Books, 1992.

[6] **L. Davis,** Handbook of Genetic Algorithms, VNR Comp. Library, 1990.

[7] **M. Dorigo, and U. Schnepf,** Genetic-based machine learning and behavior based robotics: A new synthesis, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 1, pp. 141-154, 1993.

[8] **D. Floreano, and F. Mondada,** Evolution of Homing Navigation in a Real Mobile Robot, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 26, no.3, pp. 396-407, June 1996.

[9] **D. Goldberg,** Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, 1989.

[10] **F. Hoffmann,** Evolutionary Learning of Mobile Robot Behaviors*, the First Workshop on Frontiers in Evolutionary Algorithms FEA'97*. Downloadable from the World Wide Web at http://HTTP.cs.Berkeley.EDU/~fhoffman

[11] **J. R. Koza,** Genetic Programming. A Paradigm for Genetically Breeding Populations of Computer Programs to Solve Problems*. Technical Report STAN-CS-90-1314*. Stanford University Computer Science Department, 1990.

[12] **O.Michel,** Khepera Simulator Package*,* version 2.0: Freeware mobile robot simulator written at the University of Nice Sophia-Antipolis by Olivier Michel. Downloadable from the World Wide Web at http://wwwi3s.unice.fr/~om/khep-sim.html.

[13] **S. et al Nolfi,** How to evolve autonomous robots: different approaches in evolutionary robotics, *Proceedings of the Fourth WorkShop on Artificial Life*, 1994.

[14] **J. Xiao, and L. Zhang,** Adaptive Evolutionary Planner/Navigator for Mobile Robots, *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 18-28, April 1997.